We claim:

1. Apparatus for forming an identifier for an input object and for securely marking the input object with the identifier so as to yield a marked object, the apparatus comprising:

a processor; and

a memory having computer executable instructions stored therein; and

wherein the processor, in response to the stored executable instructions:

generates a flow representation for the input object, the representation having a plurality of nodes, said nodes representing predefined first operations performed by the input object, and connections among the nodes signifying associated flow among the predefined first operations performed by the input object;

randomly selects first and second nodes from the plurality of nodes in the representation so as to form a pre-defined number of nodal pairs, each of said pairs having one of the first nodes and a corresponding one of the second nodes; and

for each of the nodal pairs, establishes flow between the first and second nodes in said each nodal pair and inserts, in the flow so established, a selected one of a plurality of different pre-defined second operations so as to collectively define the marked object, whereby the marked object implements the predefined first operations and a plurality of selected ones of the predefined second operations, each of which has been randomly spliced into flow of the input object, wherein the identifier collectively comprises all the different ones of the plurality of predefined second operations, and the associated execution flow associated therewith and involving the nodal pairs.

2. The apparatus in claim 1 wherein the input object comprises a software object.

3. The apparatus in claim 2 wherein the software object comprises, input executable code, at least one instruction in the input executable code is associated with a corresponding one of the predefined first operations, and executable code for a corresponding executable procedure is associated with each selected one of the predefined second operations.

4. The apparatus in claim 3 wherein the processor, in response to the stored instructions:

inserts a pre-defined number of separate links and designations for the selected ones of the procedures into the flow representation so as to yield a combined flow representation; and

converts, in response to said input executable code and executable code for said selected ones of the procedures, said combined flow representation into output executable code, said output executable code being the marked code.

5. The apparatus in claim 4 wherein the input executable code comprises first and second portions thereof and the flow representation comprises first and

second separate flow representations for the first and second portions of the input executable code, respectively.

6. The apparatus in claim 5 wherein:

the first portion of the input executable code comprises at least a pre-defined portion of a non-marked application program; and

the second portion of the input executable code comprises a remaining portion of the non-marked application program or pre-defined executable security code.

7. The apparatus in claim 6 wherein the processor, in response to the stored instructions, inserts executable code for the selected one procedure in noncontiguous locations in the input executable code.

8. The apparatus in claim 6 wherein the processor, in response to the stored instructions, selects the procedure from a pre-defined library of stored routines, wherein said procedure is one of the stored routines.

9. The apparatus in claim 8 wherein each of the inserted procedures implements, when executed, a pre-defined function such that if any of said inserted procedures is removed from the marked code, the marked code, when subsequently executed, will terminate its execution.

10. The apparatus in claim 8 wherein at least one of the inserted procedures implements, when executed, a pre-defined function which is independent of functionality provided by the non-marked application program.

11. The apparatus in claim 6 wherein the security code provides functionality independent of any functionality provided by the application program.

12. The apparatus in claim 6 wherein the processor, in response to the stored instructions:
(a) generates first and second separate flow representations for the first and second portions of the input executable code;
(b) partitions each of the first and second flow representations into k-clusters each so as to yield first and second cluster flow representations, respectively (where k is a pre-defined integer);
(c) randomly selects the first and second nodes in the first and second cluster flow representations, respectively, so as to form a corresponding one of the nodal pairs;
(d) inserts a designation for the selected executable procedure at a first node in the nodal pair; and
(e) repeats operations (c) and (d) a pre-defined number of times so as to insert a pre-defined number of separate procedures into the first and second flow representations so as to yield the combined flow representation.

13. The apparatus in claim 12 wherein the processor, in response to the stored instructions, inserts executable code for the selected one procedure in noncontiguous locations in the input executable code.

14. The apparatus in claim 12 wherein the processor, in response to the stored instructions, selects the one procedure from a pre-defined library of stored routines, wherein said one procedure is one of the stored routines.

15. The apparatus in claim 14 wherein each of the inserted procedures implements, when executed, a pre-defined function such that if any of said inserted procedures is removed from the marked code, the marked code, when subsequently executed, will terminate its execution.

16. The apparatus in claim 14 wherein at least one of the inserted procedures implements, when executed, a pre-defined function which is independent of functionality provided by the non-marked application program.

17. The apparatus in claim 12 wherein the processor, in response to the stored instructions:

randomly selects a node, U, in the first cluster flow representation;

randomly selects, with probability $1-\lambda$ (where $\lambda$ is a pre-defined value with $0 \leq \lambda \leq 1$), a node Y in the second cluster flow representation;

randomly selects, with probability λ, a node Z, other than U, in the first cluster flow representation; and

provides designations of nodes Y and Z as the nodes forming the nodal pair.

18. The apparatus in claim 12 wherein the processor, in response to the stored instructions, randomly selects the first and second nodes from different clusters solely within the first cluster flow representation or from different clusters solely within the second cluster flow representation.

19. The apparatus in claim 4 wherein the processor, in response to the stored instructions:
(a) partitions the flow representation into k-clusters each so as to yield a cluster flow representation;
(b) randomly selects the first and second nodes in the cluster flow representation so as to form a corresponding one of the nodal pairs;
(c) inserts the designation for the selected executable procedure at a first node in the nodal pair;
(d) repeats operations (b) and (c) a pre-defined number of times so as to insert a pre-defined number of separate procedures into the flow representation so as to yield the combined flow representation.

20. The apparatus in claim 19 wherein the processor, in response to the stored instructions, inserts executable code for the selected one procedure in noncontiguous locations in the input executable code.

21. The apparatus in claim 19 wherein the processor, in response to the stored instructions, selects the procedure from a pre-defined library of stored routines, wherein said procedure is one of the stored routines.

22. The apparatus in claim 21 wherein each of the inserted procedures implements, when executed, a pre-defined function such that if any of said inserted procedures is removed from the marked code, the marked code, when subsequently executed, will terminate its execution.

23. The apparatus in claim 21 wherein at least one of the inserted procedures implements, when executed, a pre-defined function which is independent of functionality provided by the non-marked application program.

24. The apparatus in claim 19 wherein the processor, in response to the stored instructions, randomly selects the first and second nodes from different clusters within the cluster flow representation.

25. For use with a computer system having a processor and a memory, the memory having computer executable instructions stored therein, a method for forming an identifier for input executable code and for securely marking the input executable code with the identifier so as to yield marked code, the method comprising the steps of:

generating a flow representation for the input executable code, the representation having a plurality of nodes, said nodes representing instructions in the input executable code, and connections among the nodes signifying associated control flow among instructions in the executable code;

randomly selecting first and second nodes from the plurality of nodes in the representation so as to form a pre-defined number of nodal pairs, each of said pairs having one of the first nodes and a corresponding one of the second nodes; and

for each of the nodal pairs, establishing execution flow between the first and second nodes in said each nodal pair and inserting, in the execution flow so established, executable code for a selected one of a plurality of different pre-defined executable procedures so as to collectively define the marked code, whereby the marked code contains the input executable code and a plurality of different ones of the pre-defined procedures each of which has been randomly spliced into control flow of the input executable code, wherein the identifier collectively comprises the executable code, for all the different ones of the plurality of predefined procedures, and the associated execution flows associated therewith and involving the nodal pairs.

26. The method in claim 25 wherein the input object comprises a software object.

27. The method in claim 26 wherein the software object comprises input executable code, at least one instruction

in the input executable code is associated with a corresponding one of the predefined first operation, and executable code for a corresponding executable procedure is associated with each selected one of the predefined second operations.

28. The method in claim 27 wherein the establishing and inserting step comprises the steps of:

inserting a pre-defined number of separate links and designations for the selected ones of the procedures into the flow representation so as to yield a combined flow representation; and

converting, in response to said input executable code and executable code for said selected ones of the procedures, said combined flow representation into output executable code, said output executable code being the marked code.

29. The method in claim 28 wherein the input executable code comprises first and second portions thereof and the flow representation comprises first and second separate flow representations for the first and second portions of the input executable code, respectively.

30. The method in claim 29 wherein:

the first portion of the input executable code comprises at least a pre-defined portion of a non-marked application program; and

the second portion of the input executable code comprises a remaining portion of the non-marked application program or pre-defined executable security code.

31. The method in claim 30 further comprising the step of selecting the procedure from a pre-defined library of stored routines, wherein said procedure is one of the stored routines.

32. The method in claim 31 further comprising the step of inserting executable code for the selected one procedure in noncontiguous locations in the input executable code.

33. The method in claim 31 wherein each of the inserted procedures implements, when executed, a pre-defined function such that if any of said inserted procedures is removed from the marked code, the marked code, when subsequently executed, will terminate its execution.

34. The method in claim 31 wherein at least one of the inserted procedures implements, when executed, a pre-defined function which is independent of functionality provided by the non-marked application program.

35. The method in claim 30 wherein the security code provides functionality independent of any functionality provided by the application program.

36. The method in claim 30 further comprising the steps of:

(a) generating first and second separate flow representations for the first and second portions of the input executable code;

(b) partitioning each of the first and second flow representations into k-clusters each so as to yield first and second cluster flow representations, respectively (where k is a pre-defined integer);

(c) randomly selecting the first and second nodes in the first and second cluster flow representations, respectively, so as to form a corresponding one of the nodal pairs;

(d) inserting a designation for the selected executable procedure at a first node in the nodal pair; and

(e) repeating operations (c) and (d) a pre-defined number of times so as to insert a pre-defined number of separate procedures into the first and second flow representations so as to yield the combined flow representation.

37. The method in claim 36 further comprising the step of inserting executable code for the selected one procedure in noncontiguous locations in the input executable code.

38. The method in claim 36 further comprising the step of selecting the procedure from a pre-defined library of stored routines, wherein said procedure is one of the stored routines.

39. The method in claim 38 wherein each of the inserted procedures implements, when executed, a pre-defined function such that if any of said inserted procedures is removed from the marked code, the marked code, when subsequently executed, will terminate its execution.

40. The method in claim 38 wherein at least one of the inserted procedures implements, when executed, a pre-defined function which is independent of functionality provided by the non-marked application program.

41. The method in claim 36 wherein the first and second nodes randomly selecting step comprises:

randomly selecting a node, U, in the first cluster flow representation;

randomly selecting, with probability 1-λ (where λ is a pre-defined value with $0 \leq \lambda \leq 1$), a node Y in the second cluster flow representation;

randomly selecting, with probability λ, a node Z, other than U, in the first cluster flow representation; and

providing designations of nodes Y and Z as the nodes forming the nodal pair.

42. The method in claim 36 wherein the first and second nodes randomly selecting step comprises the step of randomly selecting the first and second nodes from different clusters solely within the first cluster flow

representation or from different clusters solely within the second cluster flow representation.

43. The method in claim 28 further comprising the steps of:

(a) partitioning the flow representation into k-clusters each so as to yield a cluster flow representation;

(b) randomly selecting the first and second nodes in the cluster flow representation so as to form a corresponding one of the nodal pairs;

(c) inserting the designation for the selected executable procedure at a first node in the nodal pair; and

(d) repeating operations (b) and (c) a pre-defined number of times so as to insert a pre-defined number of separate procedures into the flow representation so as to yield the combined flow representation.

44. The method in claim 43 further comprising the step of inserting executable code for the selected one procedure in noncontiguous locations in the input executable code.

45. The method in claim 43 further comprising the step of selecting the procedure from a pre-defined library of stored routines, wherein said procedure is one of the stored routines.

46. The method in claim 45 wherein each of the inserted procedures implements, when executed, a pre-defined function such that if any of said inserted procedures is

removed from the marked code, the marked code, when subsequently executed, will terminate its execution.

47. The method in claim 45 wherein at least one of the inserted procedures implements, when executed, a pre-defined function which is independent of functionality provided by the non-marked application program.

48. The method in claim 43 wherein the first and second nodes randomly selecting step comprises the step of randomly selecting the first and second nodes from different clusters within the cluster flow representation.

49. A computer readable medium having computer executable instructions stored therein for performing the steps of claim 25.

50. Executable computer code securely marked with an identifier and generated by a computer system, the system having a processor and a memory, the memory having computer executable instructions stored therein, characterized by the code having being produced by the steps, implemented by the processor in response to the executable instructions, recited in claim 25.